# Using oracles for the design of efficient approximation algorithms

Marin Bougeret, Pierre-Francois Dutot, Denis Trystram

# Using oracles for the design of efficient approximation algorithms

Marin Bougeret (Speaker) * Pierre-Francois Dutot † Denis Trystram ‡

We are interested here in oracle techniques for the design of approximation algorithms. Following the classical definition, an oracle is a black box capable of answering correctly and instantaneously any question. Several classical $PTAS$ design techniques can be expressed using oracle formalism (by allowing the algorithm to "guess" some values during the computation).

Our objective in this work is to point out the interest of oracle techniques, beyond the design of $PTAS$. Indeed, questions to the oracle (*i.e.* guessed values) leading to non polynomial algorithms must also be considered, as the complexity may be exponential, but in a parameter that is supposed to be "small". Moreover, we aim at showing how it is possible to "degenerate" questions asked to the oracle to derive fast implementations of these interactive algorithms. These ideas will be illustrated on the classical makespan minimization on uniform machines problem ($Q||C_{max}$).

## Context : oracle algorithms

Given an instance $I$ of an optimization problem, an oracle algorithm $A_{or}$ asks the oracle for a *guess*, in the form of a string $r_I^* \in R_I$, that generally provides some information on the structure of an optimal solution. Then, the algorithm constructs a solution $A_{or}(I, r_I^*)$ for the initial problem. From such an algorithm, it is possible to derive a "classical" algorithm $A$ (without oracle), by either re-executing $A_{or}(I, r)$ for any $r \in R_I$, or constructing separately $r_I^*$ (using another algorithm). Taking the example of scheduling problems, a very classical question $r_I^*$ is an "optimal configuration" of a well-chosen small subset of $k$ tasks (among the $n$ of the instance), where $k$ is constant. Such information may allow arbitrarily good approximation ratios (like $1 + \frac{1}{k}$) at the price of subset enumeration, when simulating the oracle.

Hence, it is clear that there exist deep connections between oracle algorithms and techniques for designing approximation schemes. As shown in [1], an oracle formulation allows natural alternative definitions of several classical techniques (as those presented in [9]). Most of such techniques are based on information obtained by exhaustive enumeration or by binary search. Replacing them by oracle answers separates difficulties due to the information determination from the ones due to its utilization.

*marin.bougeret@ens-lyon.fr. Ecole Normale Superieure de Lyon, France

†pierre-francois.dutot@imag.fr. Grenoble University, France

‡trystram@imag.fr. Grenoble University and Institut Universitaire de France

## Application on the classical $Q||C_{max}$ problem

Let us consider the problem of minimizing the makespan when scheduling independent tasks on uniform machines as a case study. It is shortly denoted by $Q||C_{max}$.

Several approximation algorithms have been proposed for this problem. The 2 ratio (achieved by the classical Longest Processing Time algorithm [4]) has been improved to $\frac{3}{2}$ in [5] (using the dual approximation technique), and to 1.382-approximation in [2]. Among all existing approximation schemes, the most relevant here are the following (we list below the time complexity to achieve a ratio of $(1+\epsilon)$):

- $\mathcal{O}(mn^{\frac{10}{\epsilon^2}+3})$ in [5]
- $\mathcal{O}((\frac{1}{\epsilon}n^2)^{m-1})$ (also applies to $R||C_{max}$) in [6]
- $\mathcal{O}((n+1)^{\frac{m}{\epsilon}}poly(n,m))$ (also applies to $R||C_{max}$) in [8]
- $\mathcal{O}(n)+(\frac{log(m)}{\epsilon})^{\mathcal{O}(m^2)}$ (also applies to $R|c_{ij}|C_{max}$) in [3]
- $\mathcal{O}(2^{\mathcal{O}(1/\epsilon^2 log(1/\epsilon)^3)}poly(n,m))$ in [7]

We propose an oracle algorithm based on [5]. For any $a \in \mathbb{N}^*$, our algorithm guarantees an $1+\frac{1}{a}$ ratio by asking *some information* on the "big" tasks scheduled on each machine (*i.e.* whose computation requires more than a fraction $\frac{1}{a}$ of the total computation time on this machine).

Firstly, notice that the classical guess (*i.e.* asking the index of the big tasks scheduled on each machine) would lead to an approximation scheme with the same complexity as the one in [8]. Thus, we show how to reduce the amount of information asked, and thus the size of $R_I$, for small values of $a$ (typically $a=3$ or 4). We get for instance a $\frac{4}{3}$ (resp. a $\frac{5}{4}$)-approximation by only asking the number of big tasks for each machine, leading to an algorithm in $\mathcal{O}(2^m poly(n,m))$ (resp.$\mathcal{O}(3^m poly(n,m))$). Thus, these approximation algorithms can be faster than the better approximation schemes applied for $\epsilon$ equal to $\frac{1}{3}$ (resp. $\frac{1}{4}$), as there is no constant hidden in the exponent.

Secondly, we discuss efficient implementations where the algorithms avoid asking some sub-parts of the question. The key idea is to check if some additional *a priori* unexpected conditions become true during the execution on the particular current instance, allowing then to make a local optimal decision (without oracle query).

The approach presented in this work leads to the following natural questions for $Q||C_{max}$.

- Using only one bit of information for each machine, what information should be asked to obtain a ratio better than $\frac{4}{3}$ ?
- How to reduce the amount of information used for larger values of $a$ ?

## References


[1] M. Bougeret. Systèmes interactifs our la résolution de problèmes complexes. PhD Thesis, 2010. http://moais.imag.fr/membres/marin.bougeret/publis/these.pdf.

[2] B. Chen. Tighter bound for MULTIFIT scheduling on uniform processors. Discrete Applied Mathematics, 31(3):227–260, 1991.

[3] A.V. Fishkin, K. Jansen, and M. Mastrolilli. Grouping techniques for scheduling problems: simpler and faster. Algorithmica, 51(2):183–199, 2008.

[4] Teofilo Gonzalez, Oscar H. Ibarra, and Sartaj Sahni. Bounds for lpt schedules on uniform processors. SIAM Journal on Computing, 6(1):155–166, 1977.

[5] D. Hochbaum and D. Shmoys. A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach. SIAM Journal on Computing, 17(3):539–551, 1988.

[6] Ellis Horowitz and Sartaj Sahni. Exact and approximate algorithms for scheduling nonidentical processors. Journal of the ACM (JACM), 23(2):317–327, 1976.

[7] K. Jansen. An EPTAS for scheduling jobs on uniform processors: using an MILP relaxation with a constant number of integral variables. Automata, Languages and Programming, pages 562–573, 2009.

[8] J.K. Lenstra, D.B. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. Mathematical Programming, 46(1):259–271, 1990.

[9] P. Schuurman and G. Woeginger. Approximation schemes - a tutorial. In Lectures on Scheduling, 2000.